MA501G Diskret matematik och logik, HT 2019

Datorövning 1

På datorövningarna kommer ni att använda Matlab för att lösa problem i diskret matematik och logik. Ni bör arbeta två och två, men det är tillåtet att jobba ensam om någon absolut föredrar detta. Ni ska inte vara tre eller fler i samma grupp.

På datorövningarna kommer ni att jobba på egen hand. En övningsledare kommer att finnas till hands och svara på frågor, men eftersom ni är många så kan ni ibland få vänta en stund på er tur att få hjälp. Sitter ni fast med en speciell uppgift, fråga de som sitter bredvid om de vet vad nästa steg är, eller gå vidare och titta på ett annat problem så länge. Det viktiga är att ni inte sitter långa stunder utan aktivitet.

Under datorövning 1 ska ni bekanta er med Matlab. Målet är att ni ska bli så pass vana med Matlab att ni under de efterföljande datorövningarna är redo att ge er på lite mer komplicerade problem. En del uppgifter på datorövning 2 och 3 kommer att ingå som inlämningsuppgifter, men den här gången måste ni inte redovisa några lösningar.

Utgångspunkten för den här datorövningen är dokumentet *Introduktion i programmering med MATLAB* som ni finner på Blackboard. Om ni inte är bekanta med Matlab sedan tidigare bör ni läsa och arbeta er igenom hela introduktionen. Texten börjar från början med att beskriva vad det är vi får syn på när vi öppnar programmet. Den fortsätter sedan med att beskriva variabler, konstruktioner av loopar, etc.

1. Gör uppgifterna i *Introduktion i programmering med MATLAB*. Till dessa uppgifter behöver ni två Matlab-filer. Texten säger att ni hittar dem på Orubox, men de finns nu sparade på kursens Blackboardsida.

2. En mängd är som bekant en samling objekt som varken bryr sig om objektens ordning eller eventuella repetitioner bland dessa. En sådan oordning förekommer sällan i samband med datorer och i Matlab är de grundläggande objekten vektorer som är ordnade listor där repetition får förekomma. Ett exempel är vektorn $v = [2; 0; 1; 1]$ som innehåller de fyra talen 2, 0, 1 och 1. Trots detta kan vi för ändliga mängder av tal skriva elementen i en vektor som vi sedan kan arbeta med i Matlab. Till exempel kan mängden $U = \{1, 3, 7, 12, 107\}$ på detta sätt ”representeras” av vektorn $u = [1; 3; 7; 12; 107]$. Men observera att för Matlab är $[1; 3; 7; 12; 107]$ och $[1; 7; 3; 107; 12]$ olika objekt. Vi ska nu titta på några grundläggande funktioner som undersöker sådana här ”mängder” i Matlab.

(a) På Blackboard kan ni hitta funktionen `is_set_element`. Funktionen tar två argument, ett element och en mängd (representerad som en vektor), och testar om elementet finns i mängden. Om så är fallet ger funktionen resultatet 1, annars blir resultatet 0. Öppna funktionen i Matlab och undersök hur den är uppbyggd. Skapa några vektorer av olika storlek och testkör funktionen.[1]

(b) På Blackboard finns även den lite mer komplicerade funktionen `is_set`. Funktionen tar en vektor som argument och testar om den representerar en mängd, det vill säga om vektorn är fri från repetitioner. Öppna funktionen i Matlab och undersök hur den är uppbyggd. Skapa några vektorer av olika storlek och testkör funktionen.

(c) Skriv en Matlab-funktion `is_subset` som tar två mängder (representerade som vektorer) som argument och ger resultatet 1 om och endast om alla element i den första mängden också är element i den andra mängden. Det är tillåtet att använda ovanstående funktioner i era konstruktioner. Testkör era funktioner.

(d) Skriv en Matlab-funktion `are_equal_sets` som tar två mängder (representerade som vektorer) som argument och ger resultatet 1 om de två mängderna är lika och resultatet 0 annars. Det är tillåtet att använda ovanstående funktioner i era konstruktioner. Testkör era funktioner.

(e) Om ni har tid över, försök att skriva funktioner som tar två mängder och bestämmer snittet, unionen och mängddifferensen av dem.

[1] Här i uppgift 2 så kommer ni att uppmanas att testköra lite olika Matlab-funktioner. Exempelvis kan ni försöka hitta en övning i kursboken som ni kan besvara med hjälp av dessa funktioner.